Standard 10: Hash tables (load factor, when to apply vs other data structures)

When attempting this standard on the quiz, I did not fully understand how to solve for and explain the complexity of a hash table. I also failed to speak on and apply the load factor of the hash table.

A)

For part A, I did not specify the expected collision length for the Add, Lookup, and Remove operations. The Add expected collision length for a linked list is always 1, so for this problem the complexity of adding an item is O(1). The Lookup and Remove operations have the same complexity, O(n). This is due to us having to compare and determine if the values are the same. This is not the case with Add because we can simply insert the item into the bucket.

The load factor of this hash table would be n/1, which directly applies to how Lookup and Remove will function in this hash table. N comes from us having n items and 1 comes from us having 1 bucket.

B)

In part B, I failed again to speak on the complexities of Add, Lookup, and Remove. I also again failed to speak on and apply the load factor of the hash table.

The Add expected collision length for a linked list is always 1, so for this problem the complexity of adding an item is O(1). The Lookup and Remove operations have the same expected collision length, 1. We can solve for this by solving for the load factor of the hash table. The load factor of the table will be n/O(n). O(n) comes from us having O(n) buckets and n comes from us having n items. This load factor can be reduced to 1, so the complexity of Lookup and Remove for this hash table are O(1).

C)

In part C, I failed again to speak on the complexities of Add, Lookup, and Remove. I also again failed to speak on and apply the load factor of the hash table.

The Add expected collision length for a linked list is always 1, so for this problem the complexity of adding an item is O(1). The Lookup and Remove operations have the same expected collision length, O($\sqrt{n}$). We can solve for this by solving for the load factor of the table. The load factor of the table will be n/ O($\sqrt{n}$). O($\sqrt{n}$) comes from us having O($\sqrt{n}$) buckets and n comes from us having n items. This load factor can be reduced to O($\sqrt{n}$), so the complexity of Lookup and Remove for this hash table are O($\sqrt{n}$).

I now understand how to solve for and explain the complexity of a hash table by speaking about its relationship with the Add, Lookup, and Remove operations. I also now understand how to solve for the load factor of a hash table and why it is used when explaining the complexity of the hash table.